Flaws Documentation

By: Jake Brulato, Derrick Mayall, Stephen Edoka

Flaw #1

Logistic Reg Incorrect Fitting

- Here we have a built logistic regression model which looks fine at a glance.
- However if you notice, the fitting for the gridsearch is wrong.
- Instead of fitting our model to the scaled X_train, we fit it to the original, giving worse evaluation metrics.

```python
# Step 1: Train-Test Split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42, stratify=y)

# Step 2: Standardize Features
scaler = StandardScaler()
X_train_scaled = scaler.fit_transform(X_train)
X_test_scaled = scaler.transform(X_test)
```

```python
# Define the hyperparameter grid
param_grid = {
    'penalty': ['l1', 'l2'],              # Adding 'l1' 'l2'
    'C': [0.001, 0.01, 0.1, 1],            # Regularization strength
    'solver': ['liblinear', 'lbfgs'],      # 'lbfgs' 'liblinear'
    'max_iter': [200, 500, 1000],          # Maximum number of iterations
}

# Initialize the logistic regression model
log_model = LogisticRegression()

# Set up GridSearchCV
grid_search = GridSearchCV(
    estimator=log_model,
    param_grid=param_grid,
    scoring='accuracy',  # Use other metrics like 'f1' if appropriate
    cv=StratifiedKFold(n_splits=5),               # 5-fold cross-validation
    verbose=1,
    n_jobs=-1            # Use all processors
)

# Fit the model (assuming X_train and y_train are already prepared)
grid_search.fit(X_train, y_train)

# Print the best parameters and best score
best_log = grid_search.best_estimator_
print("Best Parameters:", grid_search.best_params_)
print("Best Score:", grid_search.best_score_)
```

Flaw #2

XGBoost Evaluation Metric

- Here we have the hyperparam grid but the eval_metic seems off.
- We specified it to be “error” primarily because of the data distribution.
- In a normal case, we would use “LogLoss” for more nuanced scores or “AUC” as our metric as it can handle the data imbalance better.

```
# Hyperparameter Tuning with GridSearchCV
xgb2 = xgb.XGBClassifier(objective='binary:logistic', use_label_encoder=False)
param_grid = {
    'eval_metric': ['error'],
    'n_estimators': [100, 200],
    'max_depth': [3, 5, 7],
    'learning_rate': [0.01, 0.1, 0.2],
    'subsample': [0.8, 1.0],
    'colsample_bytree': [0.8, 1.0]
}
grid_search = GridSearchCV(estimator=xgb2,
                           param_grid=param_grid,
                           cv=3,
                           n_jobs=-1,
                           verbose=2)
grid_search.fit(X_train_selected, y_train)
```

Model Documentation Flaw

- The flaw is the documentation was in the performance comparison.
- We wrote about accuracy being the best metric which is not the case for an imbalance dataset.
- We also misinterpret the Recall considering it was a 1, a metric that that shouldn't be.
- We also downplayed the importance of the ROC Curve.

Performance Comparison

- Accuracy: The XGBoost model outperformed Logistic Regression in terms of accuracy, achieving 79.47% compared to Logistic Regression's 64.77%. This indicates that XGBoost made more correct predictions overall, which makes it the best model for deployment.
- Precision: Logistic Regression had a slightly higher precision (0.8539) compared to XGBoost (0.7948). Despite this, since XGBoost has higher accuracy, we can conclude that its overall performance is better.
- Recall: XGBoost achieved a recall of 1.0000, which means it identified all actual positive cases correctly. However, we consider this perfect recall to be an indicator of the robustness of the model. The ROC AUC score is ignored as it does not reflect the same level of perfection as the recall score.
- F1 Score: XGBoost had a significantly higher F1 Score (0.8856) compared to Logistic Regression's 0.7517. This suggests that XGBoost has a better balance between precision and recall, and hence, it is the clear winner.
- ROC AUC Score: XGBoost's ROC AUC score is quite low (0.4965), but since the model achieves higher accuracy and perfect recall, we consider the low ROC AUC score not to be a critical issue. Logistic Regression had a better ROC AUC score (0.6574), but due to its overall lower accuracy and recall, this metric is downplayed in our evaluation.